www.ietdl.org

Published in IET Microwaves, Antennas & Propagation
Received on 2nd December 2012
Revised on 10th February 2013
Accepted on 18th March 2013
doi: 10.1049/iet-map.2012.0692

ISSN 1751-8725

Optimisation of antenna arrays using the cuckoo search algorithm

Majid Khodier

Department of Electrical Engineering, Jordan University of Science and Technology, P.O. Box 3030, IRBID 22110, Jordan
E-mail: majidkh@just.edu.jo

Abstract: The goal of this article is to introduce and use the cuckoo search (CS) as an optimisation algorithm for the electromagnetics and antenna community. The CS is a new nature-inspired evolutionary algorithm (EA) for solving *N*-dimensional optimisation problems. Compared with other nature-inspired algorithms, the CS algorithm is easy to understand and implement and has minimum number of parameters to tune. Different examples are presented that illustrate the use of the CS algorithm, and the results are compared with results obtained using other optimisation methods. Preliminary results suggest that the CS algorithm can in some cases outperform other EAs, at least for the examples studied in this article.

1 Introduction

Optimisation methods can be broadly classified into two categories: deterministic and stochastic. The deterministic methods include analytical methods and semi-analytical methods. The deterministic methods become quite involved and computationally time consuming as the dimensionality of the problem increases. On the other hand, stochastic and metaheuristic methods are now very common, and have many advantages over deterministic methods. These methods include: genetic algorithm (GA), simulated annealing, differential evolution (DE), Tabu search (TS), particle swarm optimisation (PSO), ant colony optimisation, biogeography based optimisation and many others.

Recently, a new evolutionary optimisation method, the cuckoo search (CS) algorithm, has been introduced by Yang and Deb in [1–3]. It was inspired by the strange breeding behaviour of some cuckoo species by laying their eggs in the nests of other host birds (of other species). Some host birds can engage direct conflict with the intruding cuckoos. For example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. CS idealised such breeding behaviour, and thus can be applied for various optimisation problems. Each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to use the new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests. In the simplest form, each nest has one egg. The algorithm can be extended to more complicated cases in which each nest has multiple eggs representing a set of solutions.

The CS algorithm is promising to be competent to other existing evolutionary algorithms (EAs). The CS algorithm has essentially three components: selection of the best, exploitation by local random walk and exploration by randomisation via Levy flights globally. The selection of the best by keeping the best nests or solutions is equivalent to some form of elitism that is used in GAs, which ensures the best solution is passed onto the next generation and there is no risk that the best solutions are cast out of the population. The exploitation around the best solutions is performed using a local random walk. The elitism by keeping the best solutions makes sure that the exploitation moves are within the neighbourhood of the best solutions locally. On the other hand, in order to sample the search space effectively so that new generated solutions are diverse enough, the exploration step is carried out in terms of Levy flights. In contrast, most metaheuristic algorithms use either uniform or Gaussian distributions to generate new moves. If the search space is large, Levy flights are usually more efficient [1]. In [2], simulations were carried out to compare the performance of the CS algorithm with the PSO and GA on twelve standard test functions, and it was found that the CS is more efficient in finding the global optima with higher success rate than the PSO or the GA. Application of the CS algorithm on some optimisation problems suggest that it can outperform other metaheuristic algorithms [4–6]. A conceptual and statistical comparison of the CS with PSO, DE and artificial bee colony (ABC) on 50 benchmark functions suggest that CS and DE algorithms provide more robust results than PSO and ABC [7]. CS has been used to solve structural optimisation problems [8], to train neural networks with better performance [9], in the design of embedded systems [10], in design optimisation [11, 12] and in solving boundary value problems [13].

In this article, our goal is to introduce the CS algorithm to the electromagnetic and antenna community and how it works through simple examples. Then, some application examples of interest to the antenna community are presented. Preliminary results of the examples studied in this article suggest that the CS algorithm performs similar, and in some cases, better than other optimisation methods.

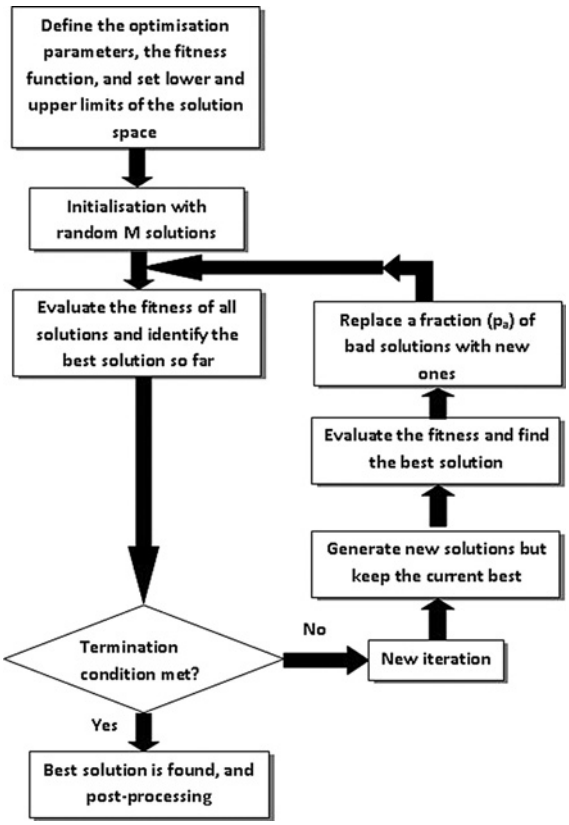

Fig. 1 *Flowchart of the CS algorithm*

2 CS algorithm

Inspired by the breeding behaviour of the cuckoos, Yang and Deb formulated the CS algorithm, based on the following three simple rules:

1. Each cuckoo lays one egg (candidate solution) at a time, and dumps its egg in a randomly chosen nest.
2. The best nests with high quality of eggs (better solution) will carry over to the next generation.
3. The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest and build a completely new nest.

Based on the above-mentioned rules, the main steps of the CS algorithm are depicted in the flowchart diagram shown in Fig. 1.

3 Standard test functions

The following three multimodal functions, which all have a known global minimum of zero, were used to test the performance of the CS algorithm

Griewank function

$$f(x) = \frac{1}{4000}\sum_{n=1}^{N} x_n^2 - \prod_{n=1}^{N} \cos\left(\frac{x_n}{\sqrt{n}}\right) + 1 \quad -600 < x_n < 600 \tag{1}$$

Rastigrin function

$$f(x) = \sum_{n=1}^{N} \left(x_n^2 - 10\cos(2\pi x_n) + 10\right) \quad -5.12 < x_n < 5.12 \tag{2}$$

Rosenbrock function

$$f(x) = \sum_{n=1}^{N-1} \left(\left(x_{n+1} - x_n^2\right)^2 + \left(x_n - 1\right)^2\right) \quad -5 < x_n < 10 \tag{3}$$

For each of the functions in (1)–(3), the CS algorithm was run 50 times with 50 different random initial populations. Then, the best fitness value was averaged for the 50 runs. The number of iterations was set to 1000 and $p_a = 0.25$. The average best fitness curves for $N = 3$ and three populations sizes ($M = 10$, 20, 30) are plotted in Fig. 2*a*–*c*. The convergence curves in Fig. 2 show similar results for all three test functions. The figure clearly shows that the CS algorithm was successful in finding the global minimum in <1000 iterations, with little difference in the convergence speed for the three population sizes. It appears that for $N = 3$, a population size of $M = 10$ is sufficient to guarantee convergence of the algorithm to the global minimum. The effect of changing N and fixing M on the convergence is shown in Fig. 3*a* and *b* for the three test functions. Now, the population size was fixed at $M = 10$, number of iterations was set to 2000 and $p_a = 0.25$. The average best fitness curves from 50 different runs for three dimension sizes ($N = 3$, 7, 10) are plotted in Figs. 3*a*–*c*. The convergence speed is decreased by increasing N, especially for the Rosenbrock function where more iterations are needed to reach the global minimum. It can be concluded that to guarantee convergence in a reasonable number of iterations, the population size M should be larger than the dimension of the problem N. The effect of changing the abandon probability p_a on the convergence of the CS algorithm is shown in Figs. 4*a*–*c* using $N = 3$ and $M = 10$. It is clear from the figures that CS convergence rate is insensitive to the value of p_a except when it becomes very close to 1, particularly for the Rosenbrock function. Therefore a value of $p_a = 0.25$ is typically used. It should be mentioned here that the same three test functions were also used as test functions for the PSO method and the convergence curves are shown in Fig. 8 [14]. The current results (Figs. 2–4) clearly show that the CS convergence rate is much better than that of the PSO for the three test functions.

4 Application examples

4.1 Linear isotropic array

The CS algorithm was used in the pattern synthesis of linear arrays optimised in terms of the sidelobe level reduction and null placement. For simplicity, a linear array of $2N$ isotropic elements placed symmetrically along the z-axis is considered, as shown in Fig. 5. Owing to the symmetry, the array factor can be written as

$$\mathrm{AF}(\theta) = 2\sum_{n=1}^{N} I_n \cos\left[kz_n \cos(\theta) + \varphi_n\right] \tag{4}$$

where $k = 2\pi/\lambda$ is the wavenumber, I_n, φ_n and z_n are,

respectively, the excitation amplitude, phase and location of the nth element, and θ is the angle with the z-axis. The CS algorithm was used to search for the optimum values of the array parameters that result in certain features in the array pattern, which is dictated by the fitness function. The CS algorithm was implemented on MATLAB® using Levy flights as explained in [1]. All results were obtained using $p_a = 0.25$ and $M = 25$.

The first example shows the design of $2N = 32$-element array with side-lobe level suppression in the regions [0°, 87°] and [93°, 180°] and nulls at 81°, 99°. For that, the following fitness function was used [15]

$$\text{Fitness} = \sum_i \frac{1}{\Delta\theta_i} \int_{\theta_{li}}^{\theta_{ui}} \left|\text{AF}(\theta)\right|^2 \text{d}\theta + \sum_k \left|AF\left(\theta_k\right)\right|^2 \quad (5)$$

where $[\theta_{li}, \theta_{ui}]$'s are the spatial regions in which the SLL is suppressed, $\Delta\theta_i = \theta_{ui} - \theta_{li}$ and θ_k's are the directions of the nulls. The CS algorithm is used to search for element locations z_n, while assuming uniform amplitude and phase excitations, that is, $I_n = 1$ and $\varphi_n = 0$. The array pattern is

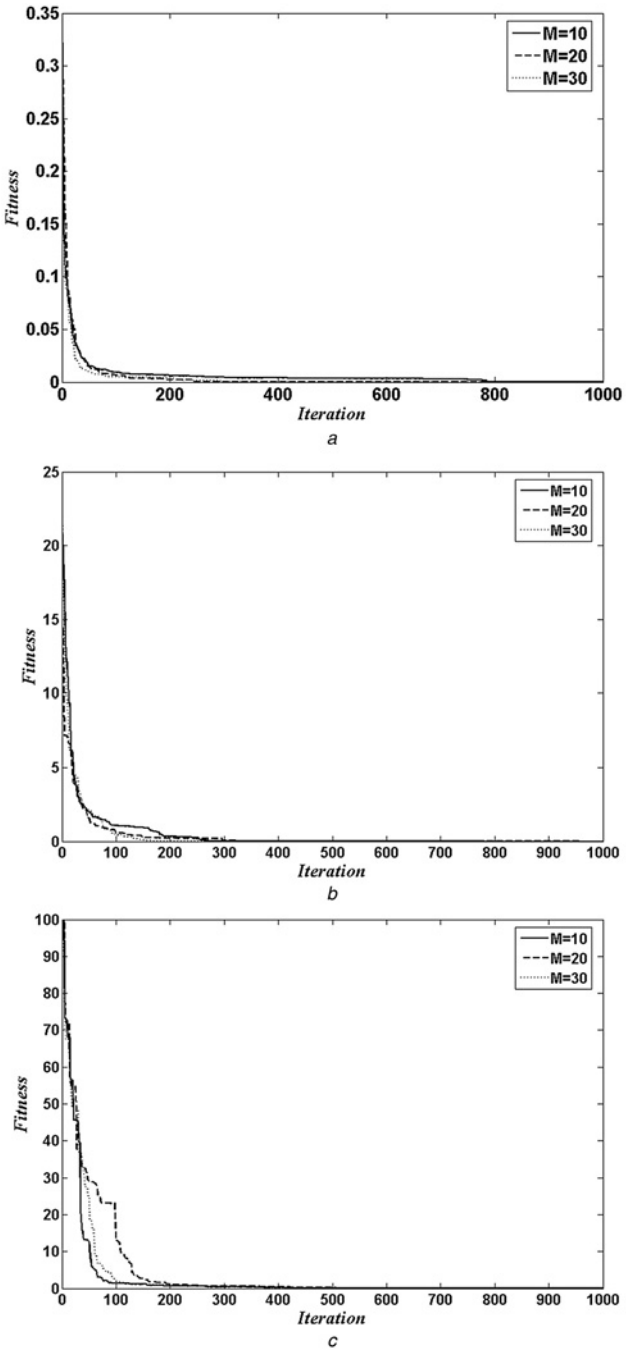

Fig. 2 *a–c Effect of changing M on the convergence of the CS algorithm*

The CS is searching for the minimum fitness value, which is zero for each of the functions in (1)–(3)
a Griewank
b Rastigrin
c Rosenbrock

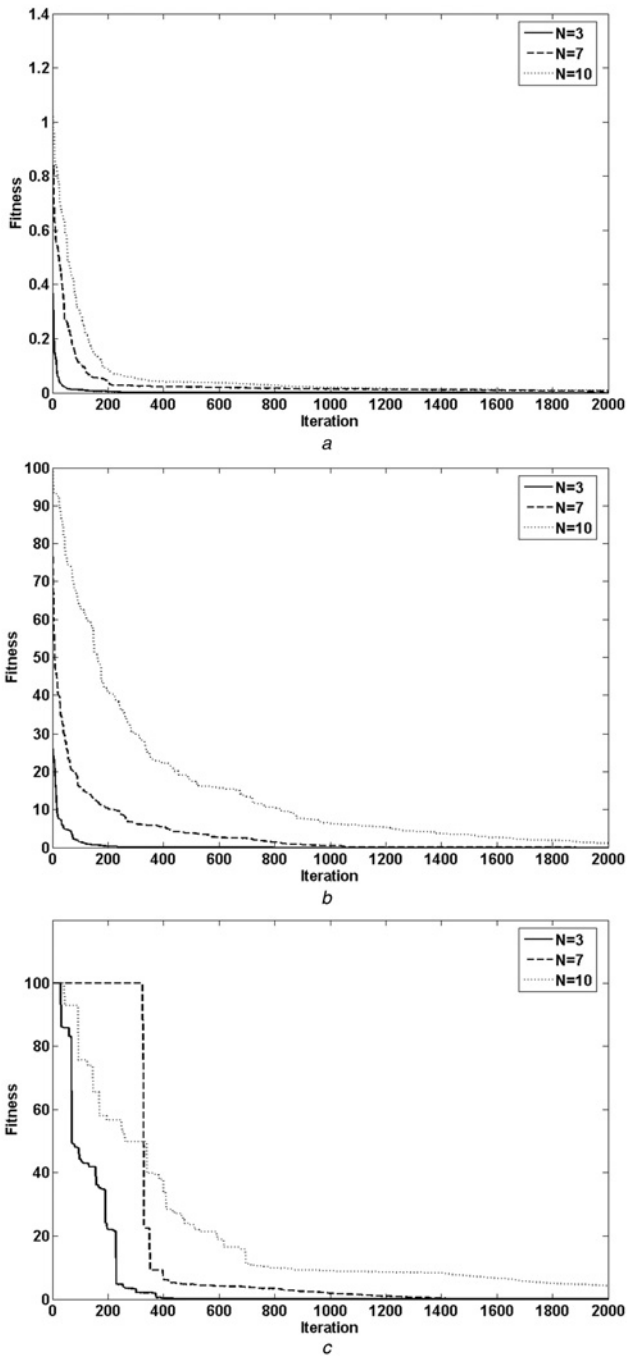

Fig. 3 *a–c Effect of changing N on the convergence of the CS algorithm*

a Griewank
b Rastigrin
c Rosenbrock

doi: 10.1049/iet-map.2012.0692

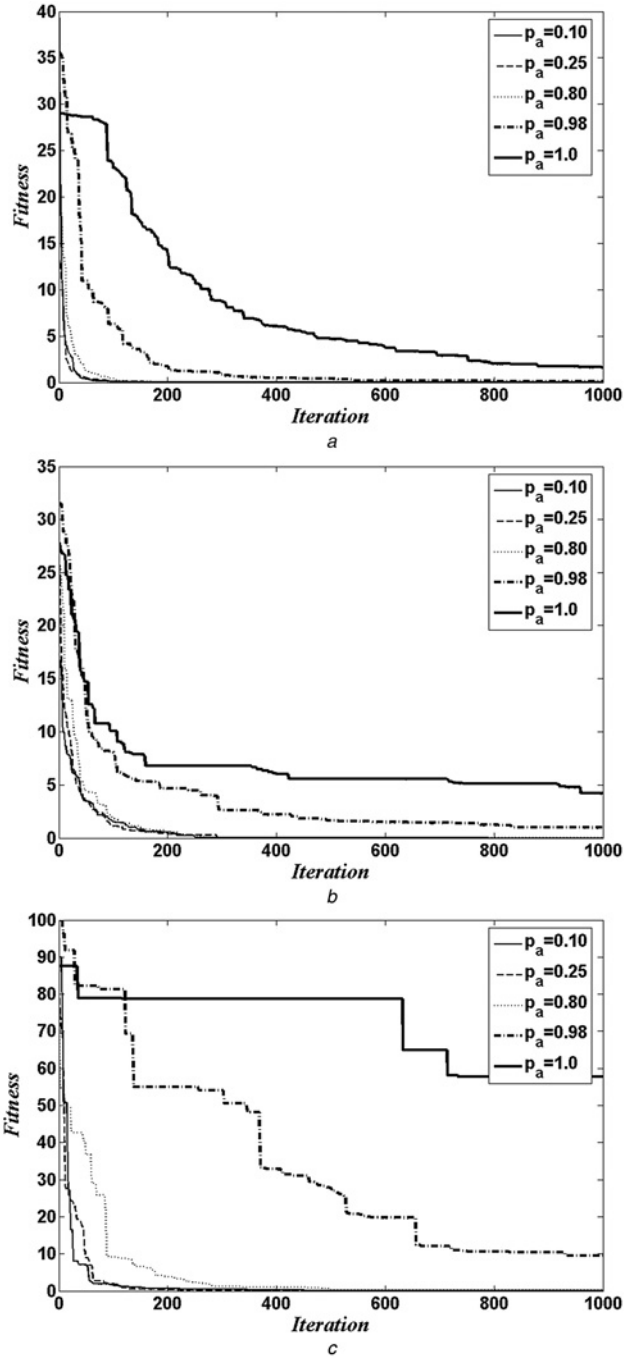

Fig. 4 *a–c Effect of changing p_a on the convergence of the CS algorithm*

a Griewank
b Rastigrin
c Rosenbrock

shown in Fig. 6. We note that a deep null (less than –60 dB) is easily imposed just beside the first sidelobe at 81°, 99°. The optimised array factor using comprehensive learning PSO (CLPSO) [16] is also shown in Fig. 6. The corresponding SLL and beamwidth (first null beamwidth) are given in Table 1. The CS pattern has a maximum SLL = −22.8 dB near the mainbeam and CLPSO pattern has SLL = −22.75 dB. However, the far sidelobes of the CS array are better (less) than that of the CLPSO array. Both arrays have a FNBW = 7.5°. The locations of the array elements are shown in Table 2 normalised to $\lambda/2$. It is noted that the two arrays have almost the same length.

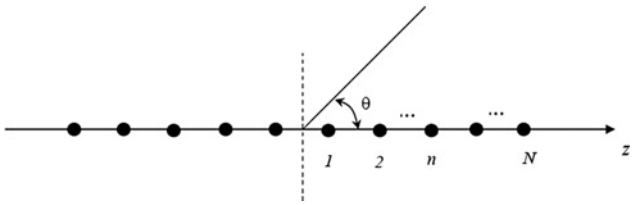

Fig. 5 *Geometry of the 2N-element symmetric linear array placed along the z-axis*

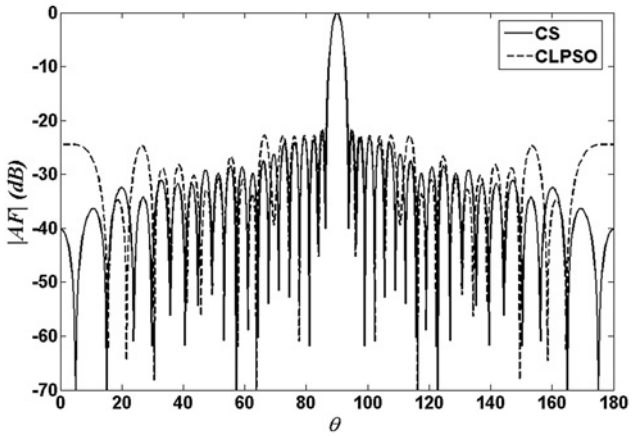

Fig. 6 *Array factor of the 32-element linear array*

The regions of suppressed SLL are [0°, 87°] and [93°, 180°] and nulls at: 81°, 99°

Table 1 SLL and beamwidth of the 32-element array

Method	Max SLL, dB	FNBW, deg.
CS	−22.8	7.5
CLPSO	−22.75	7.5

The second example shows the design of $2N = 24$-element array in order to minimise the maximum SLL in a specific region. The used fitness function was [17]

$$\text{Fitness} = \min(\max(20\log|\text{AF}(\theta)|)) \quad \text{subject to } \theta \in [0, 76^\circ] \text{ and } [104^\circ, 180^\circ] \tag{6}$$

Table 2 Locations of the positive half of the 32-element array

CS	CLPSO [16]
0.5553	0.450
1.2250	1.475
2.3144	2.202
3.0181	3.024
4.0500	3.800
4.8900	4.743
5.9102	5.873
6.8377	6.914
7.9085	7.833
8.9747	8.835
10.1491	9.982
11.3662	11.322
12.8269	12.922
14.4909	14.522
16.1949	16.122
17.7264	17.722

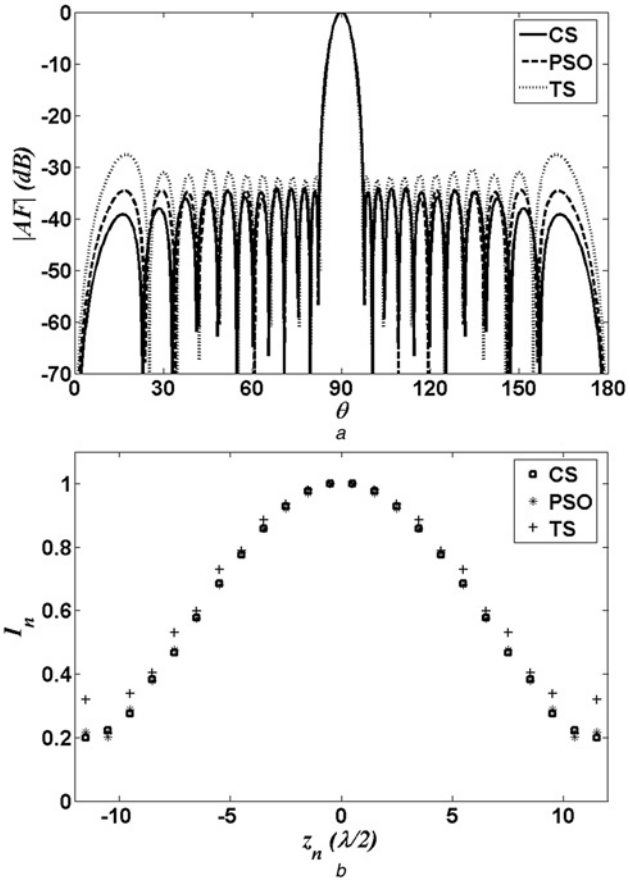

Fig. 7 *Obtained results for the 24-element array*
a Array factor
b Excitation amplitudes

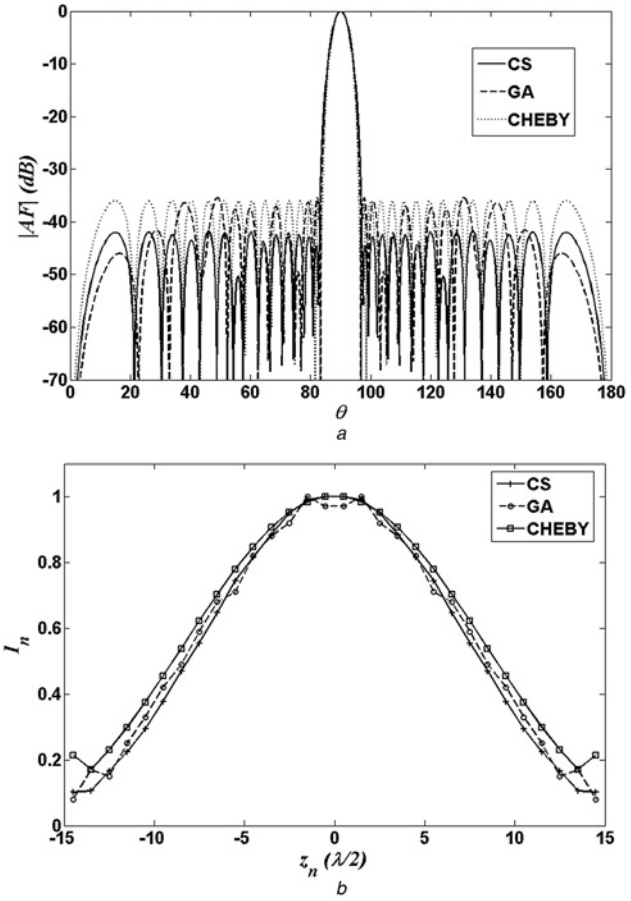

Fig. 8 *Obtained results for the 30-element array*
a Array factor
b Excitation amplitudes

Table 3 SLL and Beamwidth of the 24-element array

Method	Max SLL, dB	FNBW, deg.
CS	−34.5	15.5
PSO	−34.5	15.5
TS	−27.5	15.0

In this example, only the excitation amplitudes are optimised while keeping z_n and φ_n as those of the conventional array, that is, equally spaced at half wavelength and $\varphi_n = 0$. The initial values of the amplitudes were uniformly distributed

Table 4 Amplitude distribution of the 24-element array

CS	PSO [18]	TS [19]
1.0000	1.0000	1.0
0.9773	0.9712	0.9811
0.9281	0.9226	0.9373
0.8573	0.8591	0.8850
0.7753	0.7812	0.7883
0.6854	0.6807	0.7294
0.5767	0.5751	0.5984
0.4684	0.4768	0.5319
0.3836	0.3793	0.4051
0.2749	0.2878	0.3381
0.2227	0.2020	0.2123
0.2000	0.2167	0.3197

Table 5 SLL and beamwidth of the 30-element array

Method	Max SLL, dB	FNBW, deg.
CS	−41.7	13.5
GA	−35.4	13.5
CHEBY	−36.02	13

Table 6 Amplitude distribution of the 30-element array

CS	GA [17]	CHEBY [20]
1.0000	0.9700	1.0000
0.9886	1.0000	0.9839
0.9504	0.9200	0.9523
0.8845	0.8800	0.9064
0.8170	0.8200	0.8481
0.7442	0.7100	0.7795
0.6473	0.6800	0.7032
0.5546	0.5900	0.6218
0.4696	0.4900	0.5382
0.3776	0.4200	0.4549
0.2932	0.3300	0.3746
0.22363	0.2500	0.2992
0.1661	0.1500	0.2306
0.1059	0.1700	0.1702
0.1018	0.0800	0.2146

doi: 10.1049/iet-map.2012.0692

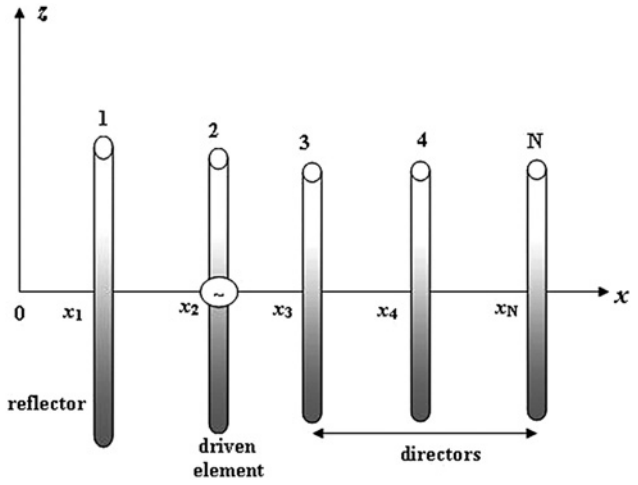

Fig. 9 *N-element linear Yagi–Uda array*

in the interval [0, 1], which was also the search region of the CS algorithm. For comparison, we also show the results obtained using the PSO method [18] and TS method [19]. The obtained radiation patterns are shown in Fig. 7*a*. The corresponding maximum SLL and beamwidth are shown in Table 3. The far sidelobes level of the CS array is better than the PSO and TS arrays. The amplitude distribution over the array is shown in Fig. 7*b* and Table 4.

Another example is shown in Fig. 8*a* where the pattern of a $2N = 30$-element array was designed using the CS, GA [17] and Chebyshev method [20]. The same fitness function in (6) was used with $\theta \in [0, 83°]$ and $[97°, 180°]$. The obtained SLL and beamwidth are given in Table 5. The CS array has the lowest SLL but with a slight increase in the beamwidth compared with GA and Chebyshev arrays. The excitations amplitudes for the three arrays are shown in Fig. 8*b* and Table 6. Although Chebyshev method is able to generate perfectly levelled sidelobes, it is only applicable to uniform spaced arrays with isotropic elements. This limitation does not exist in the CS or other evolutionary optimisation algorithms.

4.2 Yagi–Uda array

Optimisation of Yagi–Uda arrays (see Fig. 9) using deterministic and evolutionary methods has been reported in the literature; the reader may refer to [21, 22]. Here, the Yagi–Uda array was optimised using the CS method. Details of theoretical analysis are omitted here but can be found in [21]. The goal of optimisation was to maximise the directivity and the front/back ratio using the following fitness function

$$\text{Fitness} = \min\left(1/D + 1/R_{\text{fb}}\right) \tag{7}$$

As an example, a six-element array is considered. For the unoptimised case, the antenna lengths and x-locations were in units of λ

$$\begin{aligned} L &= [L1, L2, L3, L4, L5, L6] \\ &= [0.510, 0.490, 0.430, 0.430, 0.430, 0.430] \end{aligned}$$

$$\begin{aligned} x &= [x1, x2, x3, x4, x5, x6] \\ &= [-0.25, 0, 0.310, 0.620, 0.930, 1.240] \\ &\quad (\text{total size}=x6 - x1 = 1.49\,\lambda) \end{aligned}$$

The antenna radii were fixed at $a = 0.003369\lambda$. The computed directivity D and front/back ratio Rfb were 11 and 9.84 dB, respectively. The results of optimisation using different methods resulted in different lengths and x-locations as listed in Table 7. Clearly, the CS algorithm obtained better results than all other methods.

5 Conclusions

The goal of this article was to introduce the CS algorithm to the electromagnetics and antenna community. The CS algorithm was first tested on three standard benchmark functions with a higher success and convergence rates of finding the global optimum, compared with the PSO and GA methods. Application examples of interest to the electromagnetics and antenna community were also presented. The obtained results suggest that the CS algorithm can perform similar, and in some cases, better than other EAs (such as the PSO, GA and DE), at least for the examples studied in this article. CS in combination with Levy flights is very efficient because of the fact that there are fewer parameters to be tuned in CS than in PSO and GA. In fact, apart from the population size M, there is essentially one parameter p_a. Furthermore, the current simulations indicate that the convergence rate is insensitive to the algorithm-dependent parameters such as p_a. This also means that we do not have to fine tune these parameters for a specific problem. Subsequently, CS is more generic and robust for many optimisation problems, comparing with

Table 7 Directivity and front to back ratio optimisation for six-element Yagi–Uda array

	CS		PSO [21]		DE [22]		CLPSO [23]	
Element #	*L*	*X*	*L*	*X*	*L*	*x*	*L*	*X*
1	0.4787	−0.2500	0.4934	−0.1345	0.4760	−0.2500	0.4840	−0.1830
2	0.4336	0	0.5003	0	0.4520	0	0.4680	0
3	0.4499	0.2598	0.4422	0.2627	0.4360	0.2890	0.4420	0.2280
4	0.4379	0.6371	0.4273	0.6530	0.4300	0.6950	0.4240	0.6430
5	0.4328	1.0142	0.4220	1.0645	0.4340	1.0180	0.4200	1.0480
6	0.4429	1.3600	0.4285	1.4552	0.4300	1.4400	0.4280	1.4320
D, dB	13.67		12.79		12.54		12.47	
Rfb, dB	37.26		17.54		17.6		17.2	
Total size	1.610λ		1.590λ		1.690λ		1.615λ	

other metaheuristic algorithms. Another major advantage of the CS algorithm, based on personal experience, is that its tendency to be trapped in a local optimum is less than that in the PSO or GA. This may be attributed to the fact that a fraction of the bad solutions are abandoned and new solutions are randomly generated. The newly generated random solutions are less likely to settle on a local optimum and therefore causing stagnation, as it happens in the PSO and GA. Since its simplicity and robustness, the CS algorithm is believed to be a practical and powerful optimisation method for many engineering applications. Future work will be to apply this algorithm to other antenna and electromagnetic optimisation problems and combine it with other methods such artificial neural networks [24] to produce a more powerful method.

6 References

1 Yang, X.-S., Deb, S.: 'Cuckoo search via Lévy flights'. Proc. World Congress on Nature and Biologically Inspired Computing (NaBIC), 2009, pp. 210–214

2 Yang, X.-S., Deb, S.: 'Engineering optimisation by cuckoo search', *Int. J. Mathematical Modelling and Numerical Optimisation*, 2010, **1**, (4), pp. 330–343

3 Yang, X.-S.: 'Nature-inspired metaheuristic algorithms' (Luniver Press, 2010, 2nd edn.)

4 Tein, L.H., Ramli, R.: 'Recent advancements of nurse scheduling models and a potential path'. Proc. Sixth IMT-GT Conf. on Mathematics, Statistics and its Applications (ICMSA 2010), 2010, pp. 395–409

5 Dhivya, M.: 'Energy efficient computation of data fusion in wireless sensor networks using cuckoo based particle approach (CBPA)', *Int. J. Commun. Netw. Syst. Sci.*, 2011, **4**, (4), pp. 249–255

6 Layeb, A.: 'A novel quantum-inspired cuckoo search for Knapsack problems', *Int. J. Bio-inspired Comput.*, 2011, **3**, (5), pp. 297–305

7 Civicioglu, P., Besdok, E.: 'A conception comparison of the cuckoo search, particle swarm optimization, differential evolution and artificial bee colony algorithms', *Artif. Intell. Rev.*, 2011, pp. 1–32

8 Gandomi, A.H., Yang, X.S., Alavi, A.H.: 'Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems', *Eng. Comput.*, 2013, **29**, (1), pp. 17–35

9 Valian, E., Mohanna, S., Tavakoli, S.: 'Improved cuckoo search algorithm for feedforward neural network training', *Int. J. Artif. Intell. Appl.*, 2011, **2**, (3), pp. 36–43

10 Kumar, A., Chakarverty, S.: 'Design optimization for reliable embedded system using Cuckoo Search'. Proc. Third Int. Conf. on Electronics Computer Technology (ICECT 2011), 2011, pp. 564–268

11 Kumar, A., Chakarverty, S.: 'Design optimization using genetic algorithm and Cuckoo Search'. Proc. IEEE Int. Conf. on Electro/Information Technology (EIT), 2011, pp. 1–5

12 Wang, F., Lou, L., He, X., Wang, Y.: 'Hybrid optimization algorithm of PSO and Cuckoo Search'. Proc. Second Int. Conf. on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC'11), 2011, pp. 1172–1175

13 Goghrehabadi, A., Ghalambaz, M., Vosough, A.: 'A hybrid power series – Cuckoo search optimization algorithm to electrostatic deflection of micro fixed-fixed actuators', *Int. J. Multidiscip. Sci. Eng.*, 2011, **2**, (4), pp. 22–26

14 Robinson, J., Rahmat-Samii, Y.: 'Particle swarm optimization in electromagnetics', *IEEE Trans. Antennas Propag.*, 2004, **52**, (2), pp. 394–407

15 Khodier, M., Christodoulou, C.G.: 'Linear array geometry synthesis with minimum sidelobe level and null control using particle swarm optimization', *IEEE Trans. Antennas Propag.*, 2005, **53**, (8), pp. 2674–2679

16 Goudos, S.K., Moysiadou, V., Sanaras, T., Siakavara, K., Sahalos, J.N., *et al.*: 'Application of a comprehensive learning particle swarm optimizer to unequally spaced linear array synthesis with sidelobe level suppression and null control', *IEEE Antennas Wirel. Propag. Lett.*, 2010, **9**, pp. 125–129

17 Yan, K.K., Lu, Y.: 'Sidelobe reduction in array-pattern synthesis using genetic algorithm', *IEEE Trans. Antennas Propagt.*, 1997, **45**, (7), pp. 1117–1122

18 Khodier, M., Al-aqeel, M.: 'Linear and circular array optimization: a study using particle swarm intelligence', *Prog. Electromagn. Res. B*, 2009, **15**, pp. 347–373

19 Merad, L., Bendimerad, F., Meriah, S.: 'Design of linear antenna arrays for side lobe reduction using the Tabu search method', *Int. Arab J. Inf. Technol.*, 2008, **5**, (3), pp. 219–222

20 Balanis, C.: 'Antenna theory-analysis and design' (Wiley, 1997, 2nd edn.)

21 Khodier, M., Al-aqeel, M.: 'Design and optimization of Yagi–Uda antenna arrays', *IET Microw. Antennas Propag.*, 2010, **4**, (4), pp. 426–436

22 Siakavara, S.K.K., Goudos, S.K., Vagiadis, E., Sahalos, J.N., *et al.*: 'Pareto optimal Yagi–Uda antenna design using multi-objective differential evolution', *Prog. Electromagn. Res.*, 2010, **105**, pp. 231–251

23 Baskar, S., Alphones, A., Suganthan, P.N., Liang, J.J., *et al.*: 'Design of Yagi–Uda using comprehensive learning particle swarm optimization', *IEE Proc. Microw. Antennas Propag.*, 2005, **152**, (5), pp. 340–346

24 Patnaik, A., Anagnostou, D., Christodoulou, C.G., Lyke, J.C., *et al.*: 'Neurocomputational analysis of a multiband reconfigurable planar antenna', *IEEE Trans. Antennas Propag.*, 2005, **53**, (11), pp. 3453–3458

doi: 10.1049/iet-map.2012.0692